

УДК 519.612

doi: 10.15622/rcai.2025.032

**АГЕНТНО-ОРИЕНТИРОВАННЫЙ АЛГОРИТМ
РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ АЛГЕБРАИЧЕСКИХ
УРАВНЕНИЙ БОЛЬШОЙ РАЗМЕРНОСТИ**

Д.А. Берёза (*dbereza@sfedu.ru*)

Л.А. Гладков (*lagladkov@sfedu.ru*)

Н.В. Гладкова (*nvgladkova@sfedu.ru*)

Южный федеральный университет, Таганрог

В работе решается задача построения эффективного метода решения систем линейных алгебраических уравнений (СЛАУ) большой размерности. Отмечена высокая актуальность и практическая значимость решения поставленной задачи. Выполнен анализ различных подходов к построению математических моделей объектов проектирования. Приведена математическая постановка задачи. Предложен механизм снижения размерности решаемой системы уравнений на основе интеграции эволюционных алгоритмов и методов оптимизации на основе теории Парето-оптимальности. Разработан гибридный алгоритм решения СЛАУ на основе генетического поиска и многоагентной организации. Разработана архитектура программной подсистемы для решения СЛАУ на основе использования сервисно-ориентированного подхода. Выполнена программная реализация предложенного гибридного алгоритма и про-

ведены серии вычислительных экспериментов. Полученные результаты в целом подтвердили эффективность предлагаемого подхода и позволили наметить перспективные направления дальнейшего развития исследований.

Ключевые слова: системы автоматизированного проектирования, сверхбольшие интегральные схемы, многоагентные системы, сервис-ориентированные архитектуры, эволюционные вычисления.

Введение

Математическую модель объекта проектирования можно определить как множество различного рода математических элементов (переменные, упорядоченные и неупорядоченные массивы чисел и т.д.), отражающее моделируемые свойства этих элементов и взаимосвязи между ними.

Процесс построения математической модели проектируемого объекта состоит в объединении математических описаний отдельных элементов в единое целое на основе универсальных физических законов и уравнений математической физики.

В процессе автоматизированного проектирования сложного объекта, например, интегральной схемы, математическая модель формируется на основе набора уравнений множества элементарных элементов, большинство из которых являются стандартными и, следовательно, могут быть взяты из специализированных библиотек.

Сегодня на рынке представлено множество программных комплексов для схемотехнического моделирования, которые имеют один общий и весьма существенный недостаток: качество и скорость работы программной системы существенно зависит от размерности решаемой задачи. Для решения систем дифференциальных уравнений в этих программах применяются численные методы Рунге – Кутты или Гира, для систем нелинейных уравнений – метод Ньютона, а для решения систем линейных алгебраических уравнений (СЛАУ) – метод Гаусса или LU-разложение [Гридин, 2008], [Баландин и др., 2000], [Рено, 2007].

При решении реальных задач проектирования, например, проектирования сверх- и ультрабольших интегральных схем, необходимы разработка и использование новых эффективных алгоритмических и программных инструментов на всех этапах проектирования [Shervani, 1995], [Charles etc, 2009].

Актуальность данной задачи объясняется тем, что классические численные методы, которые дают удовлетворительные результаты при решении задач малой и средней размерности, не могут эффективно использоваться при проектировании СБИС в силу высокой вычислительной сложности и размерности данных задач. Это привело к появлению новых подходов к решению задач проектирования с использованием распределенных вычислений, различных метаэвристических, в том числе эволюцион-

ных и биоинспирированных, алгоритмов поиска и оптимизации [Гладков и др., 2024], [Гладков и др., 2025], [Карпенко, 2014], [Гладков и др., 2009], [Cohoon etc, 2003].

Для снижения вычислительной сложности и сокращения времени решения задач проектирования используют также алгоритмы, позволяющие распараллеливать процесс поиска, в том числе параллельные генетические алгоритмы. Также перспективным направлением является организация процесса вычислений на основе использования агентных организаций и архитектур взаимодействия [Тарасов и др., 2006], [Майоров и др., 2015], [Скобелев, 2013], [Гридин и др., 2014].

1. Постановка задачи

С точки зрения схемотехнического проектирования важнейшей задачей является создание математической модели, которая бы адекватно отображала соответствующие характеристики объекта проектирования в виде набора математических уравнений, и позволяла использовать существующие программные пакеты для решения задачи моделирования.

Любой технический объект может быть однозначно описан набором внутренних, внешних и выходных параметров. Для построения математической модели объекта, имеющего n различных характеристик используют обычно вектор внутренних параметров, каждый из которых отражают какую-либо зависимость, функцию и т.д. [Бойко и др., 2006], [Баландин, 2007].

Эффективность применения численных методов для решения задач анализа и синтеза оптимальных электрических параметров проектируемых полупроводниковых приборов в значительной мере определяется выбором начальной области, которая называется часто областью сходимости. В случае, когда местоположение начального решения выбрано неправильно и находится вне области сходимости значительно повышает риск того, что оптимальное решение не будет найдено [Влах и др., 1998].

Рост сложности проектирования, совершенствование технологий изготовления, плотности и глубины интеграции элементов привели к значительному увеличению размерности систем обыкновенных дифференциальных (ОДУ) и линейных (ЛАУ) уравнений, составляющих основу математических моделей элементной базы в микроэлектронике. Классические численные методы решения систем уравнений при проектировании не всегда обеспечивают необходимое качество и точность, кроме того, традиционные численные методы (например, градиентные методы) обладают еще одним существенным недостатком – при моделировании многоэкстремальных функций они не могут преодолевать впадины образующиеся в ландшафте функции и не способны находить выход из локальных экстремумов. Одним из возможных подходов, позволяющих снизить сложность моделирования и размерность решаемых задач является допущение различных упрощений.

$$\left\{ \begin{array}{l} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \dots + a_{1n}x_n = b_1 \\ \dots\dots\dots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \dots + a_{mn}x_n = b_m \\ a_{m+1,1}x_1 + a_{m+1,2}x_2 + a_{m+1,3}x_3 + \dots + a_{m+1,n}x_n = b_{m+1} \\ \dots\dots\dots \\ a_{2^*m,1}x_1 + a_{2^*m,2}x_2 + a_{2^*m,3}x_3 + \dots + a_{2^*m,n}x_n = b_{2^*m}, \\ \dots\dots\dots \\ a_{k^*m,1}x_1 + a_{k^*m,2}x_2 + a_{k^*m,3}x_3 + \dots + a_{k^*m,n}x_n = b_{k^*m}, \\ \dots\dots\dots \end{array} \right. \quad \left. \begin{array}{l} \text{Блок } 1 \\ \\ \text{Блок } 2 \\ \\ \text{Блок } k \end{array} \right\}$$

В результате был предложен модифицированный эволюционный алгоритм решения СЛАУ. С этой точки зрения задача заключается в нахождении решения системы линейных алгебраических уравнений:

где $A = \{a_{ij}\}$ – прямоугольная матрица коэффициентов, а x и b – искомый и заданный векторы координат $x_i, b_i \in R^n, i, j = 1, \dots, n$.

Задача состоит в нахождении оптимального значения следующей целевой функции:

$$f = \|Ax - b\| + 0.1\|x\| + F_p \rightarrow \min ,$$

где F_p – это показатель эффективности, вычисляемый по методу Парето. Схема предлагаемого эволюционного алгоритма на основе принципа Парето-оптимальности представлена на рис. 1.

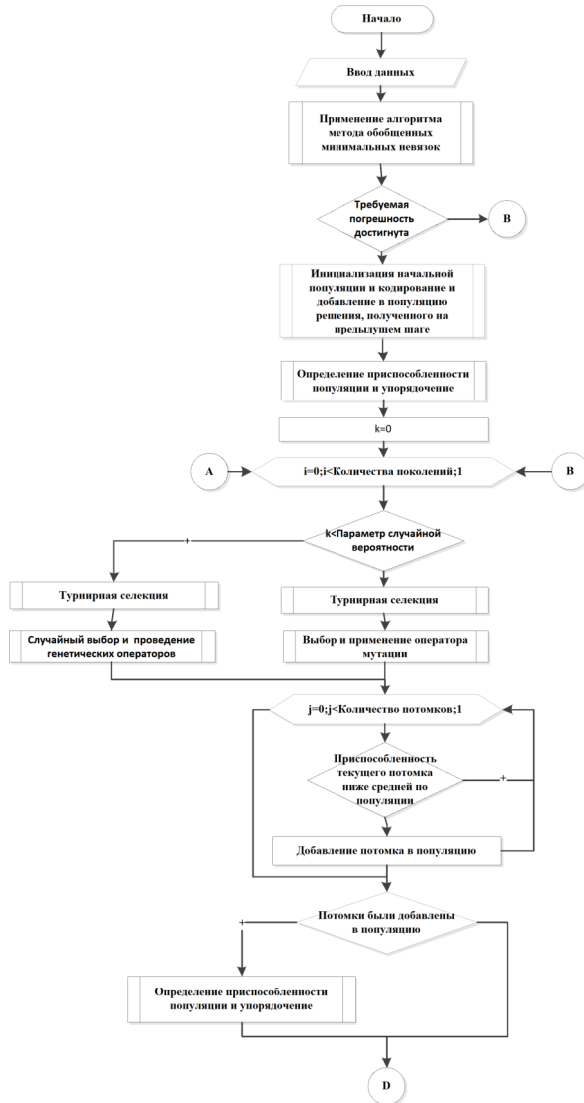


Рис. 1. Схема эволюционного алгоритма на основе принципа Парето

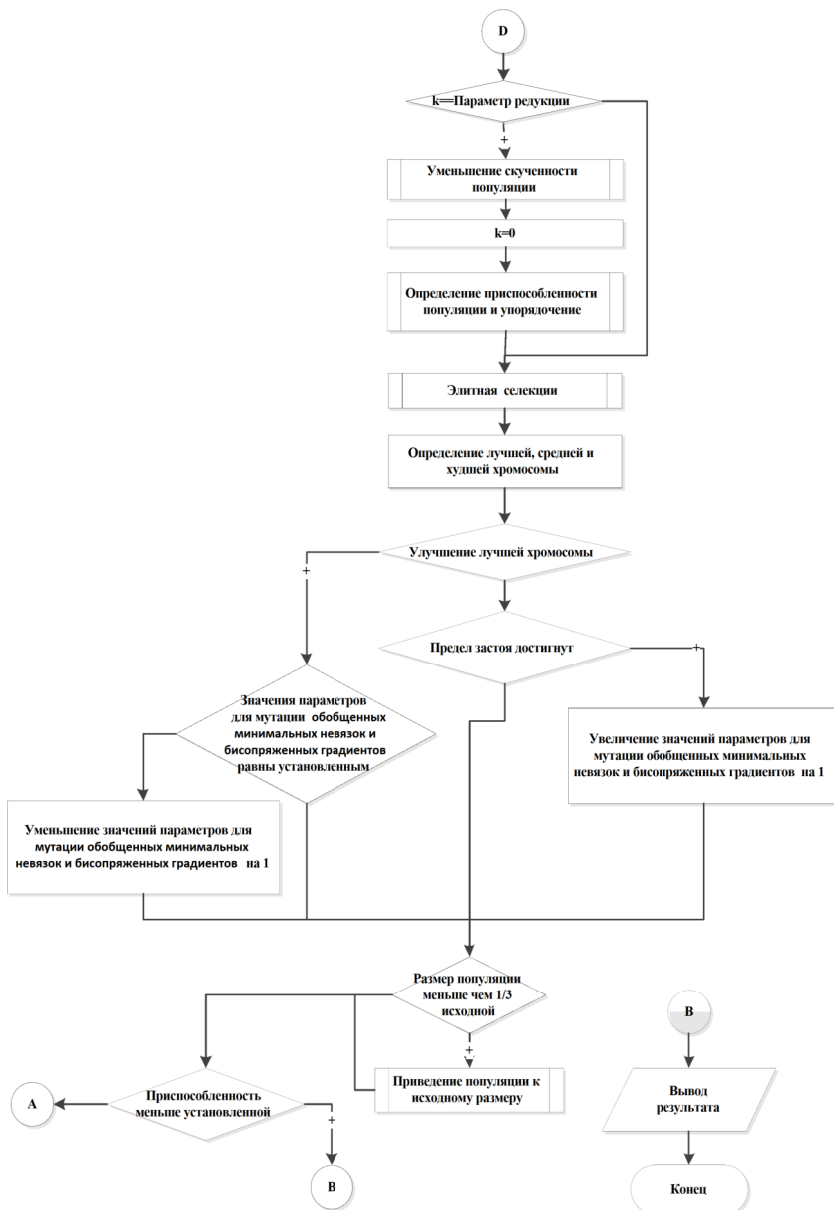


Рис. 1. Схема эволюционного алгоритма на основе принципа Парето (продолжение)

Структура предложенного алгоритма, отражающая последовательность действий, выполняемых в процессе вычислительного агента, приведена на рис. 2.

1. Φ_0 – вектор начального приближения решения;
2. $r_0 = b - A\Phi_0$;
3. \bar{r}_0 – произвольный вектор, такой, что $(\bar{r}_0, r_0) \neq 0$;
4. $\rho_0 = \alpha = \omega_0 = 1$;
5. $v_0 = p_0 = 0$;
6. для $k = 1, 2, 3, \dots$
7. $\rho_k = (\bar{r}_0, r_{k-1})$; $\beta = \left(\frac{\rho_k}{\rho_{k-1}}\right)\left(\frac{\alpha}{\omega_{k-1}}\right)$;
8. $p_k = r_{k-1} + \beta(p_{k-1} - \omega_{k-1}v_{k-1})$;
9. определение вектора u из решения системы $Mu = p_k$;
10. $v_k = Au$;
11. $\alpha = \frac{\rho_k}{(\bar{r}_0, v)}$;
12. $s = r_{k-1} - \alpha v_k$;
13. определение вектора z из решения системы $Mz = s$;
14. $t = Az$;
15. $w_k = \begin{pmatrix} t \\ z \end{pmatrix}$;
16. $\Phi_k = \Phi_{k-1} + \alpha u + \omega_k z$;
17. если Φ_k достигло требуемой точности – выход из цикла;
18. $r_k = s - \omega_k t$;
19. конец цикла по k

Рис. 2. Структура алгоритма работы вычислительного агента

2. Разработка масштабируемой мультиагентной среды

В процессе разработки архитектуры программной подсистемы было предложено использовать сервисно-ориентированный подход, в основе которого лежит идея построения подсистемы на основе взаимодействия множества различных служб (сервисов), функционирующих независимо друг от друга, каждая из которых является ответственной за реализацию одной функции. Также удобно использовать в данном случае модульный подход, суть которого состоит в обеспечении взаимозаменяемости и возможности гибко изменять функционал системы без «коренной» перестройки ее схемы. Данное свойство позволяет нам провести ассоциацию между объектами «модуль» и «агент» [Митра и др., 2003], [Baqais, 2017].

В классическом понимании термином «агент» обозначается все, что действует. Согласно принятым искусственному интеллекту и теории многоагентных систем подходе программный агент отличается от обычной компьютерной программы наличием способности к самостоятельному (автономному) функционированию в течении продолжительного периода времени, возможностью осуществлять взаимодействие с другими агентами и внешней средой, адаптироваться к изменениям и наличием цели, на достижение которой направлена его деятельность [Тарасов, 2002], [Рассел и др., 2022].

Архитектура предложенной масштабируемой многоагентной системы показана на рис. 3.

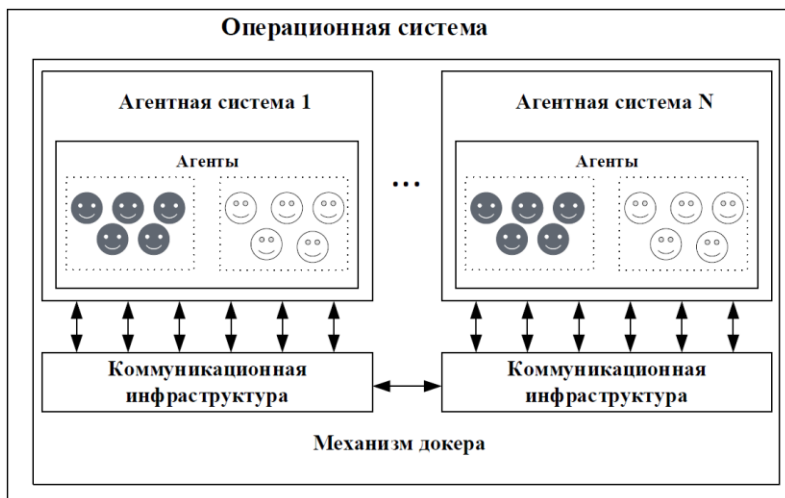


Рис. 3. Архитектура масштабируемой многоагентной системы

Контейнеризация представляет собой методологию создания программного продукта, где приложение или сервис вместе с их зависимостями и настройками (файлы, описывающие манифест развертывания) собираются в единое целое – контейнер. Такой контейнерный образ может быть протестирован как цельная единица и затем запускаться на узле операционной системы как копия созданного образа [Митра и др., 2003].

В процессе создания контейнеров и их последующей загрузки в регистры (как локальные, так и удаленные) удобно использовать инструмент Docker. Этот инструмент способствует формированию образа контейнера, включающего в себя все необходимые файлы операционной системы и приложения, а также обеспечивает его упаковку для переноса между различными платформами. Внутри контейнера обычно размещается какой-либо дистрибутив Linux, реже – упрощенная версия Windows Server.

Для инициализации процесса развертывания применяются файлы без указания расширения, имя которых совпадает с "Dockerfile", если таких файлов в каталоге только один, или же файлы с любым именем и расширением "dockerfile". Начальная строка в этих файлах традиционно содержит информацию о базовом образе (основной образ операционной системы или фреймворка), после чего следует процесс установки и конфигурирования компонентов приложения.

Docker предоставляет возможность разделения процессов сборки и доставки приложения в различных контейнерах в ходе создания образа, что отражено в одном манифесте. В таком контейнере для сборки нет необходимости в указании команды запуска. Этот подход применяется при

развертывании микросервисов на Node.js, поскольку приложения, написанные на React, требуют преобразования в оптимизированный код перед запуском. В качестве базового образа используется образ Node.js, который базируется на операционной системе Alpine Linux, специально адаптированной для эффективной работы в контейнерной среде.

Применение инструмента Docker Compose [Митра и др., 2003] облегчает процесс взаимодействия между контейнерами, их сборку и последующее их размещение. В контексте данной работы контейнеры именуются сервисами, и их параметры (включая настройки окружения, ограничения на использование ресурсов и прочее) определяются через файл `docker-compose.yml`, который является единственным в рабочей директории.

На рис. 4. представлен обобщенный агентно-ориентированный алгоритм решения больших систем линейных уравнений (СЛАУ).

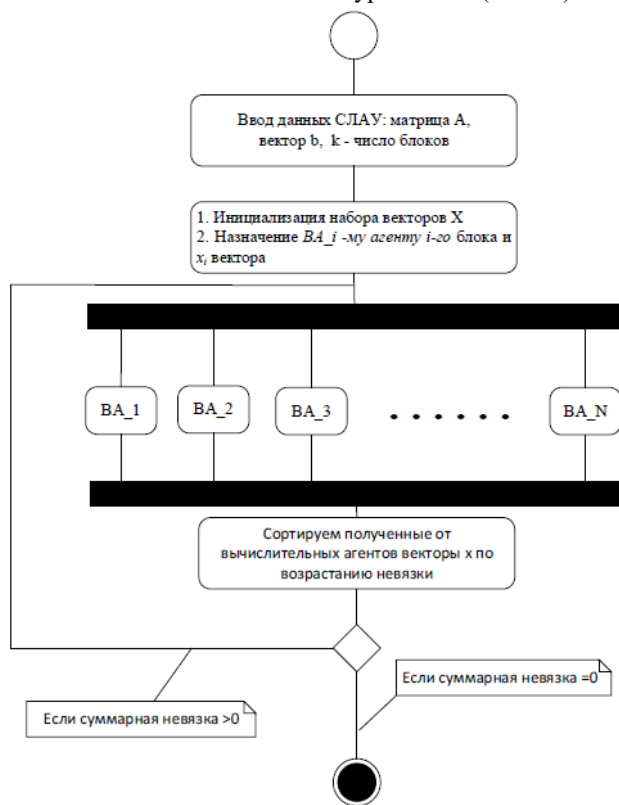


Рис. 4. Обобщенная схема алгоритма решения СЛАУ большого размера

3. Программная реализация и вычислительные эксперименты

Предложенный гибридный алгоритм был реализован в виде программной подсистемы. В процессе моделирования была поставлена задача изучить функционирование данной программной подсистемы, определить взаимозависимости между точностью и временем решения систем линейных уравнений (СЛАУ) и их размерностью, а также проверить эффективность предложенного агентно-ориентированного метода решения СЛАУ и метода решения уравнений, описывающих поведение нанометровых СБИС [Береза и др., 2025].

Критерием для оценки эффективности алгоритма является минимальное значение целевой функции, которое достигается при заданном количестве итераций и отражает точность полученного решения СЛАУ.

В ходе исследования применялась платформа для многоагентной симуляции. Эксперименты были организованы на базе тестовых систем линейных алгебраических уравнений, разработанных на этапе схемотехнического проектирования.

Для проверки эффективности разработанного алгоритма проводились серии вычислительных экспериментов для решения систем линейных алгебраических уравнений разной размерности. Для сокращения времени и объема вычислений было принято решение выбрать следующие значения параметров тестирования: доверительная вероятность – 90%; допустимая ошибка – 5%. Это позволило сократить размер экспериментальной выборки до 50. Результаты вычислительных экспериментов показаны в табл. 1.

Таблица 1
Результаты вычислительных экспериментов

№ п/п серии	Размер серии	Размер СЛАУ	Время решения (сек.)
1	50	500*500	2302
2	50	600*600	4264
3	50	700*700	4553
4	50	800*800	8123
5	50	900*900	12777

В результате выполнения статистической обработки полученных данных была построена теоретическая зависимость времени работы подсистемы от размерности решаемой задачи:

$$y = 0,0313x^2 - 16,693x + 2744,9.$$

Сравнение характера данной теоретической зависимости с результатами, полученными в ходе проведения вычислительных экспериментов, подтвердили первоначальные предположения. Характер построенных зависимостей (рис. 5) позволяет сделать однозначный вывод о том, что временная сложность разработанного гибридного алгоритма является полиномиальной.

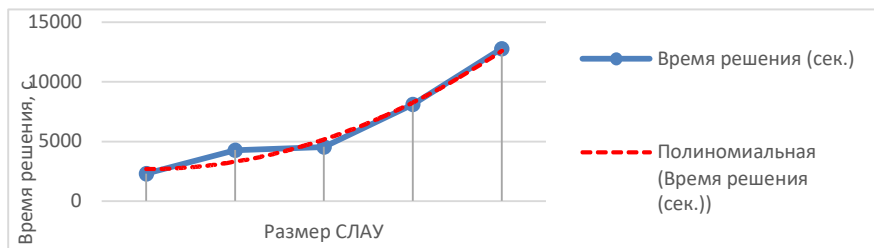


Рис. 5. Зависимость времени решения от размера СЛАУ

Заключение

На основании полученных экспериментальных данных можно сделать вывод о том, что разработанный гибридный алгоритм может эффективно использоваться для решения систем линейных алгебраических уравнений, имеет квадратичную временную сложность и демонстрирует стабильность функционирования вне зависимости от роста размерности задачи. В дальнейшем планируется продолжить исследования в направлении повышения эффективности предлагаемого подхода за счет использования биоинспирированных методов поиска.

Список литературы

- [Баландин и др., 2000] Баландин М.Ю., Шурина Э.П. Методы решения СЛАУ большой размерности. – Новосибирск: Изд-во НГТУ. 2000.
- [Береза и др., 2025] Береза Д.А., Гладков Л.А., Гладкова Н.В. Разработка агентно-ориентированного алгоритма решения систем линейных алгебраических уравнений большой размерности. // Известия ЮФУ. Технические науки. – 2025. – № 1. – С. 22-37.
- [Бойко и др., 2006] Бойко А.Я., Безруков А.Е., Русаков А.С., Ткачев Д.Ф., Хапаев М.М. Новый алгоритм вычисления двумерных емкостей в задаче экстракции емкости. // II Всероссийская научно-техническая конференция «Проблемы разработки перспективных микроэлектронных систем»: Сб. научных тр. под общ. ред. А.Л. Стемповского. – М.: ИПИМ РАН, 2006.
- [Влах и др., 1998] Влах И., Сингхал К. Машинные методы анализа и проектирования электронных схем: пер. с англ. – М.: Радио и связь, 1988.

- [Гладков, 2000] Гладков Л.А. О некоторых подходах к построению гибридных интеллектуальных систем для решения графовых задач // Новости искусственного интеллекта. – 2000. – № 3. – С. 71-90.
- [Гладков и др., 2009] Гладков Л.А., Курейчик В.В., Курейчик В.М., Сороколетов П.В. Биоинспирированные методы в оптимизации. – М.: Физматлит, 2009.
- [Гладков и др., 2011] Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы. – М.: Физматлит, 2011.
- [Гладков и др., 2024] Гладков Л.А., Кравченко Ю.А., Курейчик В.В., Родзин С.И. Интеллектуальные системы: модели и методы метаэвристической оптимизации. Монография. – Чебоксары: Среда, 2024.
- [Гридин, 2008] Гридин В.Н. Численно-аналитическое моделирование радиоэлектронных схем. – М.: Наука, 2008.
- [Гридин и др., 2014] Гридин В.Н., Дмитриевич Г.Д., Анисимов Д.А. Архитектура распределенных сервис-ориентированных систем автоматизированного проектирования // Известия ЮФУ. Технические науки. – 2014. – № 7(156). – С. 51-58.
- [Карпенко, 2014] Карпенко А.П. Современные алгоритмы поисковой оптимизации. Алгоритмы, вдохновленные природой. – М.: МГТУ им. Баумана, 2014.
- [Майоров и др., 2015] Майоров И.В., Скобелев П.О. Модель динамики агентов потребностей и возможностей // Труды XVII Международной конференции «Проблемы управления и моделирования в сложных системах», Самара, 2015. – Самара: СНЦ РАН, 2015. – С. 79-87.
- [Митра и др., 2003] Митра Р., Надарешвили И. Микросервисы. От архитектуры до релиза. – СПб.: Питер, 2023.
- [Потапов, 2007] Потапов Ю. Технология экстракции паразитных параметров для моделирования межсоединений // Технологии в электронной промышленности. – 2007. – № 6. – С. 22-26.
- [Рассел и др., 2022] Рассел С., Норвиг П. Искусственный интеллект: современный подход. – М.: Издательский дом «Вильямс», 2022.
- [Рено, 2007] Рено Н.Н. Численные методы. – М.: КДУ, 2007.
- [Скобелев, 2013] Скобелев П.О. Ситуационное управление и мультиагентные технологии: коллективный поиск согласованных решений в диалоге // Онтологии проектирования. – 2013. – № 2(8). – С. 26-48.
- [Тарасов, 2002] Тарасов В.Б. От многоагентных систем к интеллектуальным организациям. – М.: Эдиториал УРСС, 2002.
- [Тарасов и др., 2006] Тарасов В.Б., Голубин А.В. Эволюционное проектирование: на границе между проектированием и самоорганизацией // Известия ТРТУ. Технические науки. – 2006. – № 8(63). – С. 77-82.
- [Baqais, 2017] Baqais A.A.B. A Multi-view Comparison of Various Metaheuristic and Soft Computing Algorithms // I.J. Mathematical Sciences and Computing. – 2017. – 1.3 (4). – P. 8-19.
- [Charles etc, 2009] Alpert, Charles J., Mehta, Dinesh P., Sapatnekar, Sachin S. Handbook of algorithms for physical design automation. – CRC Press, New York, USA, 2009.
- [Cohon etc, 2003] Cohoon, J.P., Karro, J., Lienig, J. Evolutionary Algorithms for the Physical Design of VLSI Circuits. Advances in Evolutionary Computing: Theory and Applications / Ghosh, A., Tsutsui, S. (eds.) – Springer Verlag, London, 2003.
- [Shervani, 1995] Shervani N. Algorithms for VLSI physical design automation. – USA, Kluwer Academy Publisher, 1995.